Media Over QUIC
Internet-Draft
Intended status: Standards Track
Expires: 20 April 2026

S. Nandakumar
Cisco
C. Jennings
Cisco Systems

17 October 2025

Media over QUIC Transport (MOQT) for Agent-to-Agent Protocol
draft-nandakumar-a2a-moqt-transport-00

Abstract

This document specifies how the Agent-to-Agent (A2A) protocol can be transported over Media over QUIC Transport (MOQT). MOQT provides a publish/subscribe model over QUIC that enables efficient real-time communication between AI agents, supporting the A2A protocol's requirements for discovery, negotiation, and collaboration while leveraging MOQT's streaming capabilities and built-in prioritization mechanisms.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://ietf-wg-moq.github.io/draft-a2a-moqt-transport/draft-a2a-moqt-transport.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-a2a-moqt-transport/.

Discussion of this document takes place on the Media Over QUIC Working Group mailing list (mailto:moq@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/moq/. Subscribe at https://www.ietf.org/mailman/listinfo/moq/.

Source for this draft and an issue tracker can be found at https://github.com/ietf-wg-moq/draft-a2a-moqt-transport.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 20 April 2026.

Copyright Notice

Copyright (c) 2025 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction 3
  1.1. Requirements Language 3
2. A2A Transport Requirements 3
3. MOQT Transport Mapping 4
  3.1. Connection Establishment 4
  3.2. Track Organization 5
  3.3. Message Serialization 5
  3.4. Priority and Quality of Service 6
4. Protocol Operations 6
  4.1. Agent Discovery 6
  4.2. Request/Response Patterns 7
  4.3. Streaming Operations 8
5. Benefits of MOQT for A2A 9
  5.1. Real-time Communication Benefits 9
  5.2. Scalability Benefits 10
  5.3. Reliability and Resilience 10
6. MOQT Relay Infrastructure for A2A 10
  6.1. Relay Network Architecture 10
  6.2. Message Caching Benefits 11
7. References 11
  7.1. Normative References 11
  7.2. Informative References 12
Appendix A. Acknowledgments 12
Authors' Addresses 12

1. Introduction

The Agent-to-Agent (A2A) protocol enables communication between AI agents across different platforms and frameworks. Currently, A2A supports three transport protocols: JSON-RPC 2.0 over HTTP(S), gRPC, and HTTP+JSON/REST. This document defines a fourth transport option using Media over QUIC Transport (MOQT) to provide enhanced real-time streaming capabilities for agent interactions.

MOQT is designed for media streaming over QUIC and WebTransport, providing a publish/subscribe model with hierarchical data organization. While originally intended for media applications, MOQT's streaming model and prioritization capabilities make it well-suited for real-time agent communication scenarios.

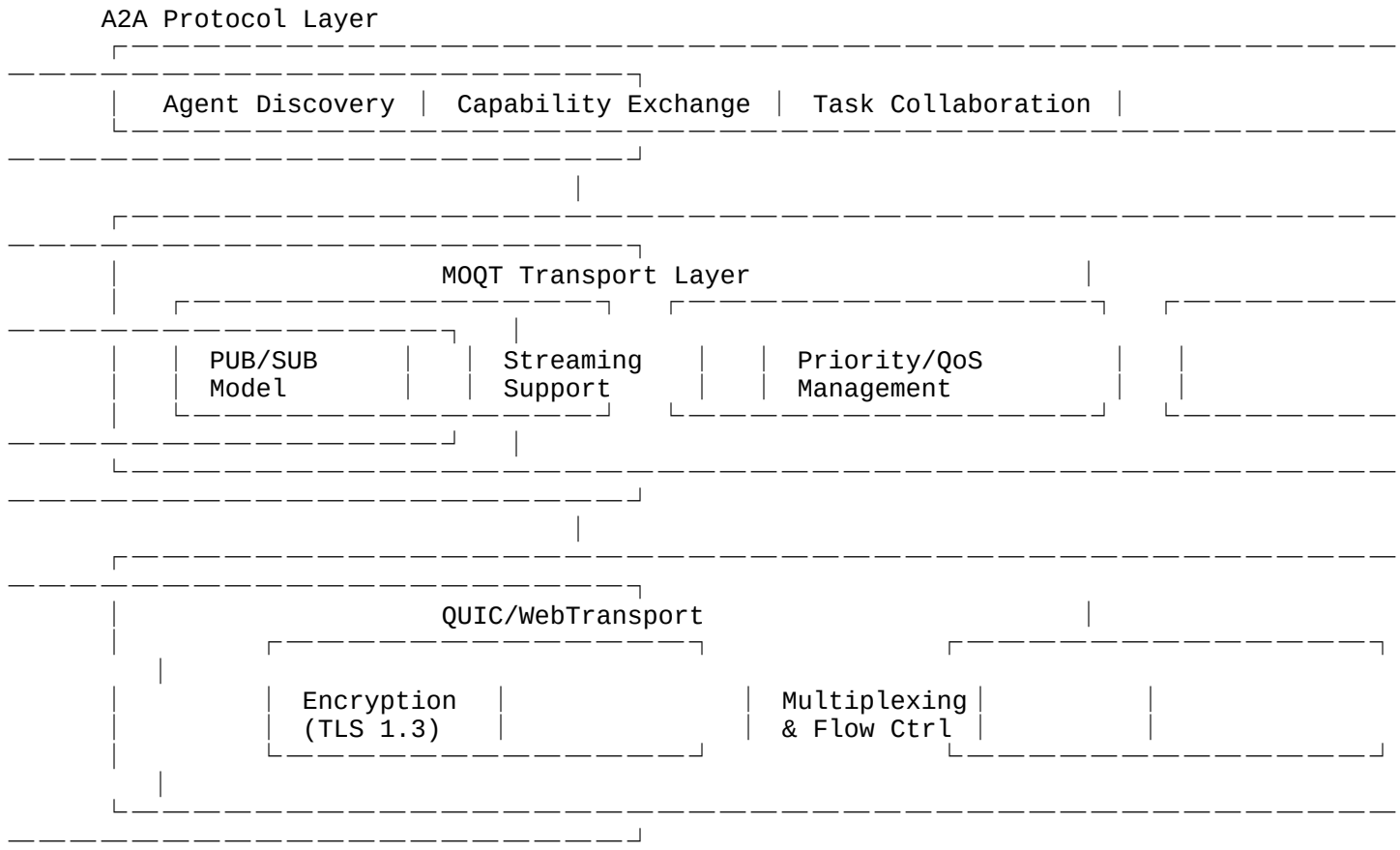
1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. A2A Transport Requirements

According to the A2A specification, all transport protocols MUST provide functional equivalence and support the following capabilities:

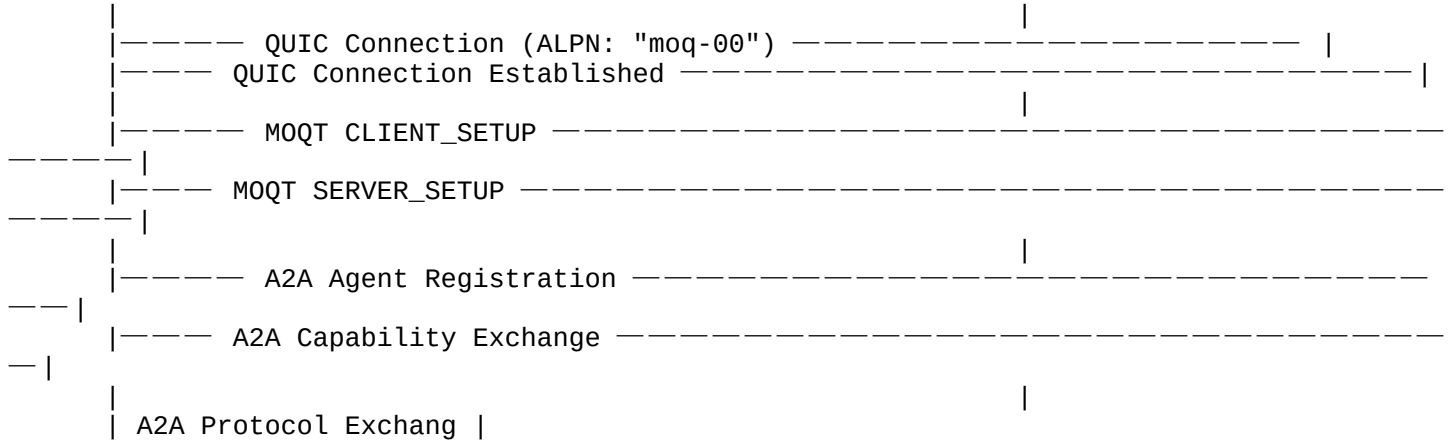
* Secure communication over encrypted channels
* Request/response messaging patterns
* Streaming data delivery capabilities
* Error handling and status reporting
* Agent discovery and capability negotiation
* Support for various data types (text, JSON, files)



3. MOQT Transport Mapping

3.1. Connection Establishment

A2A agents using MOQT transport MUST establish a QUIC or WebTransport connection with the ALPN value "moq-00" as defined in the MOQT specification. The connection setup follows this sequence:



During the MOQT setup phase, agents MUST negotiate the following A2A-specific extensions:

* a2a-version: Supported A2A protocol version

* a2a-capabilities: Agent capabilities and supported operations

* a2a-auth: Authentication and authorization tokens

3.2. Track Organization

A2A protocol messages are mapped to MOQT tracks using a hierarchical namespace structure. Each agent creates tracks for different message categories: Tracks are organized as follows:

Request Tracks: Used for A2A request messages, organized by category and action (e.g., discovery/capabilities, tasks/execute).

Response Tracks: Used for A2A response messages, organized by request ID for correlation.

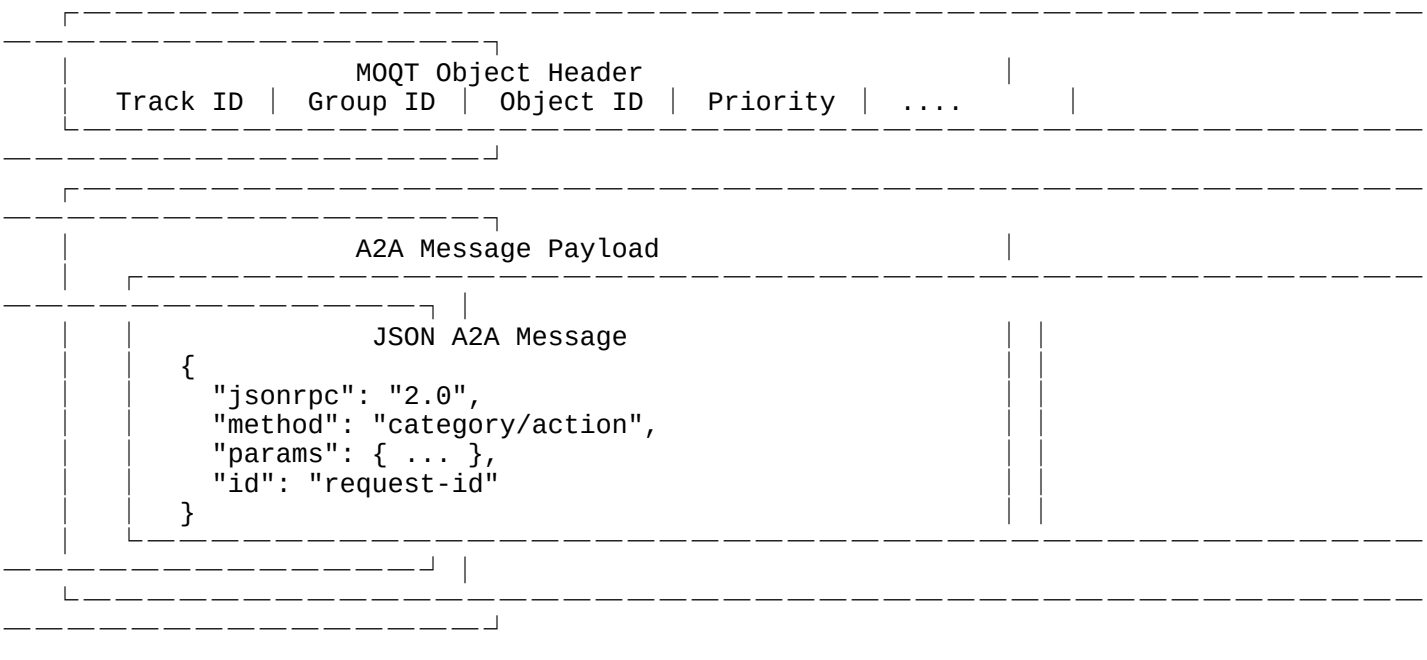
Stream Tracks: Used for long-running streaming operations like task execution updates or file transfers.

Notification Tracks: Used for asynchronous notifications and events.

3.3. Message Serialization

A2A protocol messages are serialized as JSON and encapsulated within MOQT objects. Each MOQT object contains:

MOQT Object Structure for A2A:



The A2A message format remains unchanged from the JSON-RPC 2.0 specification, ensuring compatibility with existing A2A implementations.

3.4. Priority and Quality of Service

MOQT's built-in priority system is mapped to A2A message urgency:

Priority 1 (Highest): Emergency and critical error responses

Priority 2: Real-time interaction responses and notifications

Priority 3: Standard request/response messages

Priority 4: Discovery and capability exchange messages

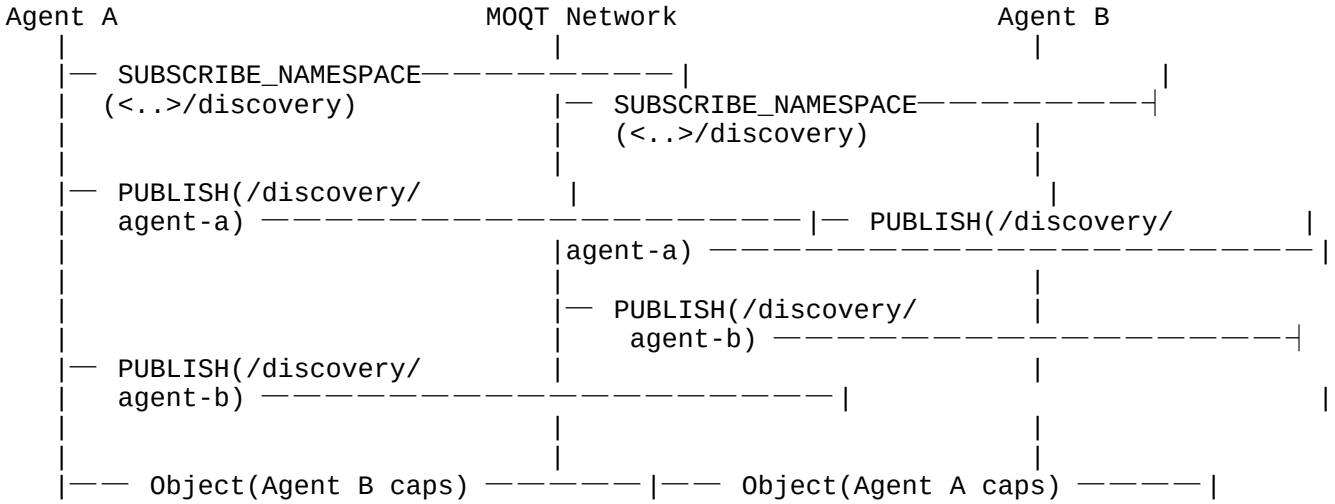
Priority 5 (Lowest): Background updates and non-critical notifications

4. Protocol Operations

4.1. Agent Discovery

Agent discovery over MOQT uses a Subscribe Namespace pattern where agents subscribe to the <session_context_id>/discovery namespace prefix and publish their agent announcements using the <session_context_id>/discovery/agent-{id} pattern to reach all subscribers.

Discovery Flow:



4.2. Request/Response Patterns

A2A request/response interactions are implemented using coordinated PUBLISH/SUBSCRIBE operations:

1. Server agent uses SUBSCRIBE_NAMESPACE to subscribe to <..>/server-agent-{id} Namespace prefix

2. Client agent sends PUBLISH message on <..>/server-agent-{id}/request track

3. Client agent subscribes to <..>/server-agent-{id}/response track

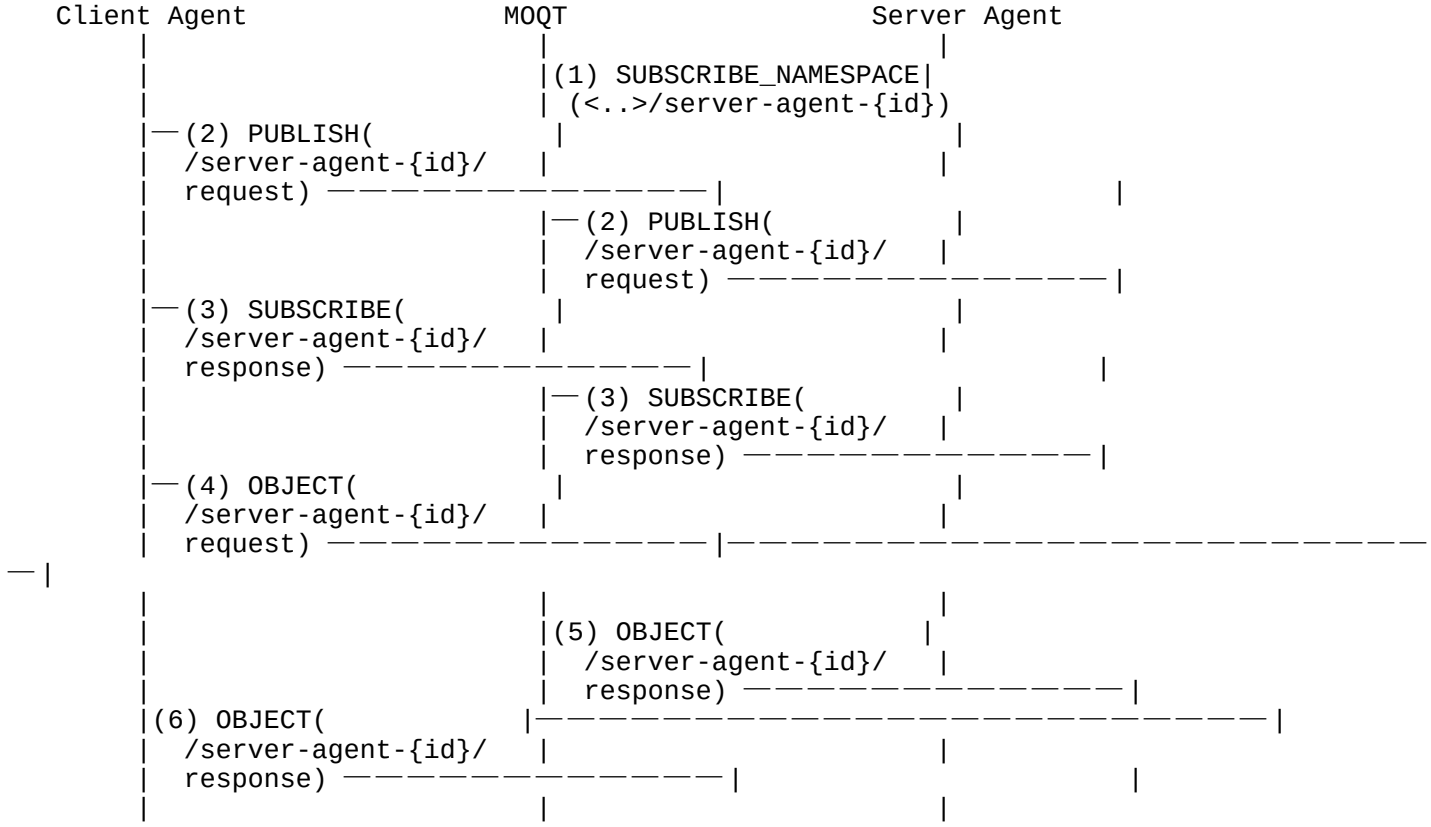
4. Client agent publishes OBJECT on <..>/server-agent-{id}/request track

5. Server agent receives OBJECT via its namespace subscription

6. Server agent publishes OBJECT on <..>/server-agent-{id}/response track

7. Client agent receives response OBJECT via its subscription

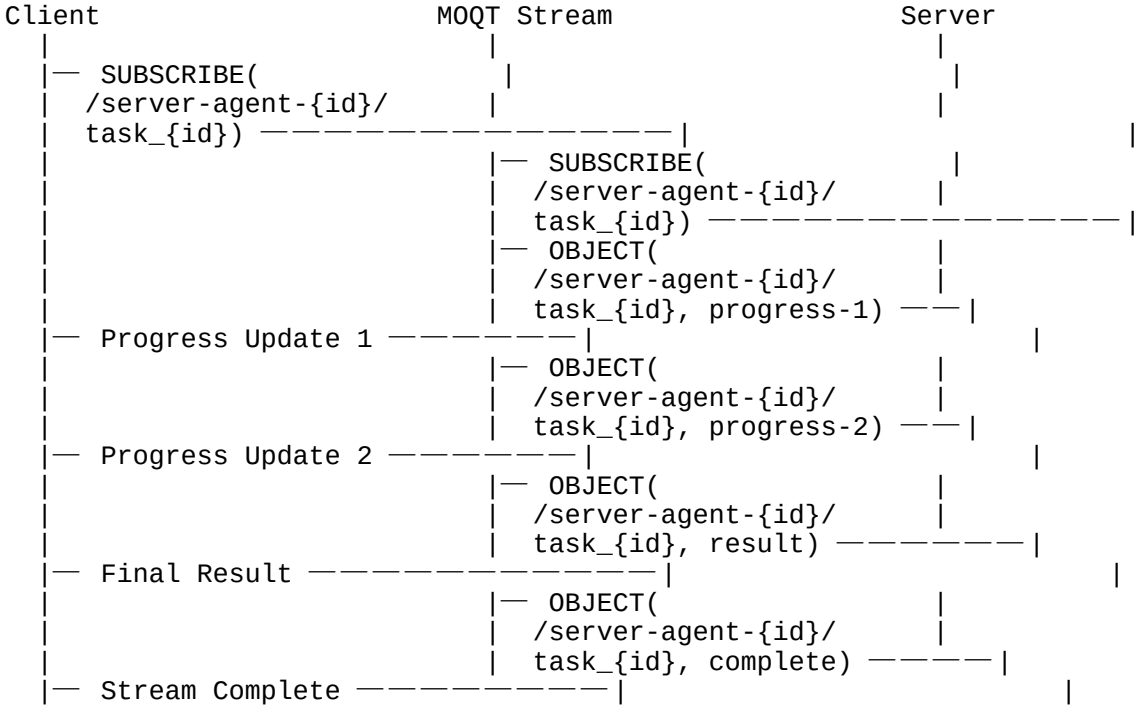
Request/Response Flow:



4.3. Streaming Operations

Long-running A2A operations like task execution or file transfer utilize MOQT's streaming capabilities. Stream tracks carry incremental updates as separate MOQT objects.

Streaming Task Execution:



5. Benefits of MOQT for A2A

MOQT provides significant advantages over traditional A2A transports, particularly for real-time agent collaboration and large-scale deployments:

5.1. Real-time Communication Benefits

* Low-latency streaming enables immediate agent response to changing conditions
* Priority-based message delivery ensures critical communications are not delayed
* Streaming object delivery allows agents to act on incomplete data when appropriate
* Built-in flow control prevents agent overload during high-volume interactions

5.2. Scalability Benefits

MOQT's publish/subscribe model creates powerful network effects for multi-agent environments:

* One-to-many broadcasts enable efficient agent coordination
* Subscription-based interest management reduces unnecessary traffic
* Dynamic agent discovery scales to thousands of participants
* Relay-based distribution prevents connection explosion in mesh networks

5.3. Reliability and Resilience

MOQT provides enhanced reliability features critical for production agent deployments:

* Connection migration maintains agent sessions during network changes
* Automatic retry and recovery mechanisms handle transient failures
* Graceful degradation allows agents to continue operating with reduced capabilities
* Quality of Service controls prevent cascade failures in agent networks

6. MOQT Relay Infrastructure for A2A

MOQT relays provide critical infrastructure benefits for large-scale A2A deployments, enabling efficient message distribution and caching.

6.1. Relay Network Architecture

MOQT relays form a hierarchical distribution network that optimizes A2A message delivery across geographic and organizational boundaries:

This hierarchical structure provides:

* Regional optimization with local agent clusters
* Global connectivity for cross-region agent collaboration
* Load distribution to prevent single points of failure

* Configurable routing policies based on agent requirements

6.2. Message Caching Benefits

MOQT relays can implement intelligent caching strategies optimized for A2A communication patterns.

The most frequently accessed and time-sensitive data with time-to-live values are measured in seconds. This includes real-time agent responses, live data streams, and status updates that require immediate availability. These messages are kept in high-speed memory to ensure minimal latency for active agent interactions.

Another layer can serve as an intermediate storage tier with time-to-live values measured in minutes. This layer contains agent capability profiles, metadata about agent configurations, and results from common queries that are accessed regularly. The caching layer can provide a balance between accessibility and resource utilization.

Also for archives of historical data with extended retention periods measured in hours or days, can employ a different storage strategy. Examples data might include historical logs, archived computation results, compliance audit data, and debug traces that may be needed for analysis or regulatory purposes but are accessed infrequently.

7. References

7.1. Normative References

[A2A] A2A Project, "Agent-to-Agent Protocol Specification", 2025, <https://a2a-protocol.org/latest/specification/>.

[MoQ-TRANSPORT] Nandakumar, S., Vasiliev, V., Swett, I., and A. Frindell, "Media over QUIC Transport", Work in Progress, Internet-Draft, draft-ietf-moq-transport-14, 2 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-moq-transport-14>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

7.2. Informative References

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <https://www.rfc-editor.org/info/rfc7540>.

[WebTransport] Vasiliev, V., "The WebTransport Protocol Framework", RFC 9297, August 2023, <https://www.rfc-editor.org/rfc/rfc9297>.

Appendix A. Acknowledgments

Authors' Addresses

Suhas Nandakumar
Cisco
Email: snandaku@cisco.com

Cullen Jennings
Cisco Systems
Email: fluffy@cisco.com